Process Opacity and Insertion Functions

Damas P. Gruska[1], M. Carmen Ruiz[2]

[1]Comenius University, Slovak Republic
[2]Universidad de Castilla-La Mancha, Spain

Abstract
Time insertion functions as a way how to guarantee state-based security with respect to timing attacks are proposed and studied. As regards the security property, we work with the property called process opacity. First, we define timing attacks and later we show how security with respect to them can be enforced by such functions. The time insertion function can alter the time behaviour of the original system by inserting some time delays to guarantee its security. We investigate conditions under which such functions do exist and also some of their properties.

Keywords
state-based security, process opacity, process algebras, information flow, insertion function, timing attack

1. Introduction

Formal methods allows us, in many cases, to show, even prove, that a given system is not secure. Then we have a couple of options what to do. We can either re-design its behavior, what might be costly, difficult or even impossible, in the case that it is already part of a hardware solution, proprietary firmware and so on, or we can use supervisory control (see [1]) to restrict system's behaviour in such a way that the system becomes secure. A supervisor can see (some) system's actions and can control (disable or enable) some set of system's action. In this way it restricts system's behaviour to guarantee its security (see also [2]). This is a trade-off between security and functionality. Situation is different in the case of timing attacks. They, as side channel attacks, represent serious threat for many systems. They allow intruders "break" "unbreakable" systems, algorithms, protocols, etc. For example, by carefully measuring the amount of time required to perform private key operations, attackers may be able to find fixed Diffie-Hellman exponents, factor RSA keys, and break other cryptosystems (see [3]). This idea was developed in [4] were a timing attack against smart card implementation of RSA was conducted. In [5], a timing attack on the RC5 block encryption algorithm is described. The analysis is motivated by the possibility that some implementations of RC5 could result in data-dependent rotations taking a time that is a function of the data. In [6], the vulnerability of two implementations of the Data Encryption Standard (DES) cryptosystem under a timing attack is studied. It is showed that a timing attack yields the Hamming weight of the key used by both DES implementations. Moreover, the attack is computationally inexpensive. A timing attack against an implementation of AES candidate Rijndael is described in [7], and the one against the popular SSH protocol in

29th International Workshop on Concurrency, Specification and Programming (CS&P'21)
damas.gruska@fmph.uniba.sk (D. P. Gruska); MCarmen.Ruiz@uclm.es (M. C. Ruiz)

[8]. Even relatively recently discovered possible attacks on most of currently used processors (Meltdown and Spectre) also belong to timing attacks. To protect systems against timing attacks we propose application of inserting functions (see [9, 10, 11, 12]). Such functions can add some idling between actions to enforce process's security. In this paper we investigate conditions under which such functions do exist and also their properties.

As regards formalism, we will work with a timed process algebra and opacity, which is the security property based on an absence of information flow. This formalism enables us to formalize timing attacks. In [13] we have introduced time insertion functions to guarantee language-based security and showed some of their properties. In [14] we studied conditions under which there exists a timed insertion function for a given process and security language-based security property and we have presented some decidability and undecidability results. In this paper we define and study time insertion functions for state-based security property process opacity.

The paper is organized as follows. In Section 2 we describe the timed process algebra TPA which will be used as a basic formalism and information flow state-based security process opacity. The next section is devoted to time insertion functions as a way how to guarantee state this security property with respect to timing attacks.

2. Working Formalism

In this section we briefly recall Timed Process Algebra, TPA for short (for more details see [15]). TPA is based on Milner's Calculus of Communicating Systems (for short, CCS, see [16]) but the special time action t which expresses elapsing of (discrete) time is added and hence the set of actions is extended from Act to $Actt$. The presented language is a slight simplification of Timed Security Process Algebra (tSPA) introduced in [17]. We omit an explicit idling operator ι used in tSPA and instead of this we allow implicit idling of processes. Hence processes can perform either "enforced idling" by performing t actions which are explicitly expressed in their descriptions or "voluntary idling" (i.e. for example, the process $a.Nil$ can perform t action despite the fact that this action is not formally expressed in the process specification). But in both cases internal communications have priority to action t in the parallel composition. Moreover we do not divide actions into private and public ones as it is in tSPA. TPA differs also from the tCryptoSPA (see [18]). TPA does not use value passing and strictly preserves *time determinancy* in case of choice operator $+$ what is not the case of tCryptoSPA (see [15]).

To define the language TPA, we first assume a set of atomic action symbols A not containing symbols τ and t, and such that for every $a \in A$ there exists $\overline{a} \in A$ and $\overline{\overline{a}} = a$. We define $Act = A \cup \{\tau\}, At = A \cup \{t\}, Actt = Act \cup \{t\}$. We assume that $a, b, \dots$ range over A, $u, v, \dots$ range over Act, and $x, y \dots$ range over $Actt$.

We give a structural operational semantics of terms by means of labeled transition systems. The set of terms represents a set of states, labels are actions from $Actt$. The transition relation $\rightarrow$ is a subset of TPA $\times$ $Actt$ $\times$ TPA. We write $P \xrightarrow{x} P'$ instead of $(P, x, P') \in \rightarrow$ and $P \stackrel{x}{\not\rightarrow}$ if there is no P' such that $P \xrightarrow{x} P'$. The meaning of the expression $P \xrightarrow{x} P'$ is that the term P can evolve to P' by performing action x, by $P \xrightarrow{x}$ we will denote that there exists a term P' such that $P \xrightarrow{x} P'$. We define the transition relation as the least relation satisfying the inference rules for CCS plus the following inference rules:

$$\frac{}{Nil \xrightarrow{t} Nil} \quad A1 \qquad \frac{}{u.P \xrightarrow{t} u.P} \quad A2$$

$$\frac{P \xrightarrow{t} P', Q \xrightarrow{t} Q', P \mid Q \not\xrightarrow{\tau}}{P \mid Q \xrightarrow{t} P' \mid Q'} \quad Pa \qquad \frac{P \xrightarrow{t} P', Q \xrightarrow{t} Q'}{P + Q \xrightarrow{t} P' + Q'} \quad S$$

For $s = x_1.x_2.\dots.x_n, x_i \in Act$ we write $P \xrightarrow{s}$ instead of $P \xrightarrow{x_1}\xrightarrow{x_2} \dots \xrightarrow{x_n}$ and we say that s is a trace of P. The set of all traces of P will be denoted by $Tr(P)$. By ϵ we denote the empty sequence and by M^* we denote the set of sequences of elements from M. We use $\overset{x}{\Rightarrow}$ for transitions including τ actions (see [16]). By $s|_B$ we will denote the sequence obtained from s by removing all actions not belonging to B. By $L(P)$ we will denote a set of actions which can be performed by P, i.e. $L(P) = \{x | P \xrightarrow{s.x}, s \in Actt^*\}$.

We define two behavior equivalences trace equivalence and weak bisimulation, respectively (see [16]).

Definition 1. *The set of weak traces of process P is defined as $Tr_w(P) = \{s \in At^\star | \exists P'.P \overset{s}{\Rightarrow} P'\}$.*

Two processes P and Q are weakly trace equivalent iff $Tr_w(P) = Tr_w(Q)$.

Definition 2. *Let $(TPA, Act, \rightarrow)$ be a labelled transition system (LTS). A relation $\Re \subseteq TPA \times TPA$ is called a* weak bisimulation *if it is symmetric and it satisfies the following condition: if $(P, Q) \in \Re$ and $P \xrightarrow{x} P', x \in Actt$ then there exists a process Q' such that $Q \overset{\widehat{x}}{\Rightarrow} Q'$ and $(P', Q') \in \Re$. Two processes P, Q are weakly bisimilar, abbreviated $P \approx Q$, if there exists a weak bisimulation relating P and Q.*

To formalize an information flow we do not divide actions into public and private ones at the system description level, as it is done for example in [18], but we use a more general concept of observation and opacity. This concept was exploited in [19] and [20] in a framework of Petri Nets and transition systems, respectively. Firstly we define observation function on sequences from $Act^\star$. Various variants of observation functions differs according to contexts which they take into account. For example, an observation of an action can depend on the previous actions.

Definition 3 (Observation). *Let Θ be a set of elements called observables. Any function $\mathcal{O} : Actt^\star \rightarrow \Theta^\star$ is an observation function. It is called static /dynamic /orwellian / m-orwellian $(m \geq 1)$ if the following conditions hold respectively (below we assume $w = x_1 \dots x_n$):*

- *static if there is a mapping $\mathcal{O}' : Actt \rightarrow \Theta \cup \{\epsilon\}$ such that for every $w \in Act^\star$ it holds $\mathcal{O}(w) = \mathcal{O}'(x_1) \dots \mathcal{O}'(x_n)$,*
- *dynamic if there is a mapping $\mathcal{O}' : Actt^\star \rightarrow \Theta \cup \{\epsilon\}$ such that for every $w \in Actt^\star$ it holds $\mathcal{O}(w) = \mathcal{O}'(x_1).\mathcal{O}'(x_1.x_2) \dots \mathcal{O}'(x_1 \dots x_n)$,*
- *orwellian if there is a mapping $\mathcal{O}' : Actt \times Actt^\star \rightarrow \Theta \cup \{\epsilon\}$ such that for every $w \in Actt^\star$ it holds $\mathcal{O}(w) = \mathcal{O}'(x_1, w).\mathcal{O}'(x_2, w) \dots \mathcal{O}'(x_n, w)$,*

- *m-orwellian if there is a mapping $\mathcal{O}' : Actt \times Actt^\star \to \Theta \cup \{\epsilon\}$ such that for every $w \in Actt^\star$ it holds $\mathcal{O}(w) = \mathcal{O}'(x_1, w_1).\mathcal{O}'(x_2, w_2) \ldots \mathcal{O}'(x_n, w_n)$ where $w_i = x_{max\{1,i-m+1\}}.x_{max\{1,i-m+1\}+1} \cdots x_{min\{n,i+m-1\}}$.*

In the case of the static observation function each action is observed independently from its context. In the case of the dynamic observation function an observation of an action depends on the previous ones, in the case of the orwellian and m-orwellian observation function an observation of an action depends on the all and on m previous actions in the sequence, respectively. The static observation function is the special case of m-orwellian one for $m = 1$. Note that from the practical point of view the m-orwellian observation functions are the most interesting ones. An observation expresses what an observer - eavesdropper can see from a system behavior and we will alternatively use both the terms (observation - observer) with the same meaning. Note that the same action can be seen differently during an observation (except static observation function) and this express a possibility to accumulate some knowledge by intruder. For example, an action not visible at the beginning could become somehow observable. From now on we will assume that $\Theta \subseteq Actt$.

Now let us assume hat an intruder is interested whether a given process has reached a state with some given property which is expressed by a (total) predicate. This property might be process deadlock, capability to execute only traces with some given actions, capability to perform at the same actions form a given set, incapacity to idle (to perform τ action) etc. We do not put any restriction on such predicates but we only assume that they are consistent with some suitable behavioral equivalence. The formal definition follows.

Definition 4. *We say that the predicate ϕ over processes is consistent with respect to relation $\cong$ if whenever $P \cong P'$ then $\phi(P) \Leftrightarrow \phi(P')$.*

As the consistency relation $\cong$ we could take bisimulation, weak bisimulation, weak trace equivalence or any other suitable equivalence.

An intruder cannot learn validity of predicate ϕ observing process's behaviour iff there are two traces, undistinguished for him (by observation function $\mathcal{O}$), where one leads to a state which satisfy ϕ and another one leads to a state for which $\neg\phi$ holds. The formal definition follows.

Definition 5 (Process Opacity). *Given process P, a predicate ϕ over processes is process opaque w.r.t. the observation function $\mathcal{O}$ whenever $P \xrightarrow{w} P'$ for $w \in Actt^*$ and $\phi(P')$ holds then there exists P'' such that $P \xrightarrow{w'} P''$ for some $w' \in Actt^*$ and $\neg\phi(P'')$ holds and moreover $\mathcal{O}(w) = \mathcal{O}(w')$. The set of processes for which the predicate ϕ is process opaque w.r.t. to the $\mathcal{O}$ will be denoted by $POp^{\phi}_{\mathcal{O}}$.*

3. Insertion functions

Timing attacks belong to powerful tools for attackers who can observe or interfere with systems in real time. On the other side these techniques is useless for off-line systems and hence they could be consider safe with respect to attackers who cannot observe (real) time behaviour. By the presented formalism we have a way how to distinguish these two cases. First we define untimed version of an observation function, i.e. a function which does not take into account time information. From now on we will consider observation functions $\mathcal{O} : Actt^\star \to Actt^\star$ for which there exists untimed variants $\mathcal{O}_t$. Function $\mathcal{O}_t$ is untimed variant of $\mathcal{O}$ iff $\mathcal{O}(w) = \mathcal{O}_t(w|_{Act})$, i.e. untimed variant represents an observer who does not see elapsing of time since both traces, with and without actions t, are seen equally. Now we can formally describe situation when a process could be jeopardized by a timing attack i.e. is secure only if an observer cannot see elapsing of time.

Definition 6 (Timinig Attacks). *We say that process P is prone to timing attacks with respect to ϕ and $\mathcal{O}$ iff $P \notin POp^{\phi}_{\mathcal{O}}$ but $P \in POp^{\phi}_{\mathcal{O}_t}$.*

Example 1. *Let us assume an intruder who tries to learn whether a private action h was performed by a given process while (s)he can observe only public action l but not h itself. Then process $P = a.t.R + b.t.t.Q$ is not opaque for static observation function $\mathcal{O}(x) = \epsilon$ for $x \neq t$, $\mathcal{O}(t) = t$ and $\phi(R), \neg\phi(Q)$ hold, i.e. $P \notin POp^{\phi}_{\mathcal{O}}$. But if an observer cannot see elapsing of time this process is opaque, i.e. $P \in POp^{\phi}_{\mathcal{O}_t}$.*

From now on we will consider only processes which are prone to timing attacks (see Definition 6) and moreover we will assume that predicate ϕ is decidable. There are basically three ways how to solve vulnerability to timing attacks except putting a system off-line. First, redesign the system, put some monitor or supervisor which prevents dangerous behavior which could leak some classified information (see, for example, [2]) or hide this leakage by inserting some time delays between system's action (see [12, 10] for general insertion functions for non-deterministic finite automata). Now we will define and study this possibility. First we need some notation. For $w, w' \in Actt^*$ and $w = x_1.x_2 \dots x_n$ we will write $w \ll_S w'$ for $S \subset Actt$ iff $w' = x_1.i_1.x_2 \dots x_n.i_n$ where $i_k \in S^*$ for every $k, 1 \leq k \leq n$. In general, an insertion function inserts additional actions between original process's actions (given by trace w) such that for the resulting trace w' we have $w \ll_S w'$ and w' is still a possible trace of the process. In our case we would consider insertion functions (called time insertion functions) which insert only time actions i.e. such functions that $w \ll_{\{t\}} w'$. Results of an insertion function depends on previous process behaviour. We can define this dependency similarly as it is defined for observation functions.

Definition 7 (Time Insertion function). *Any function $\mathcal{F} : Actt^\star \to Actt^\star$ is an insertion function iff for every $w \in Actt^*$ we have $w \ll_{\{t\}} \mathcal{F}(w)$. It is called static /dynamic /orwellian / m-orwellian ($m \geq 1$) if the following conditions hold respectively (below we assume $w = x_1 \dots x_n$):*

- *static if there is a mapping $f : Actt \to \{t\}^*$ such that for every $w \in Actt^\star$ it holds $\mathcal{F}(w) = x_1.f(x_1).x_2.f(x_2) \dots x_n.f(x_n)$,*
- *dynamic if there is a mapping $f : Actt^\star \to \{t\}^*$ such that for every $w \in Act^\star$ it holds $\mathcal{F}(w) = x_1.f(x_1).x_2.f(x_1.x_2) \dots x_n.f(x_1. \dots .x_n)$,*
- *orwellian if there is a mapping $f' : Actt \times Actt^\star \to \{t\}^*$ such that for every $w \in Actt^\star$ it holds $\mathcal{F}(w) = x_1.f(x_1, w).x_2.f(x_2, w) \dots x_n.f(x_n, w)$,*

- *m-orwellian if there is a mapping* $f' : Actt \times Actt^\star \to \{t\}^*$ *such that for every* $w \in Actt^\star$ *it holds* $\mathcal{F}(w) = x_1.f(x_1, w_1).x_2.f(x_2, w_2) \ldots x_n.f(x_n, w_n)$, $w_i = x_{max\{1,i-m+1\}}.x_{max\{1,i-m+1\}+1} \ldots x_{min\{n,i+m-1\}}$.

Note that contrary to general insertion function (see [12, 10]) inserting time actions is much simpler due to transition rules $A1, A2, S$. The purpose of time insertion function is to guaranty security with respect to process opacity. Let $P \notin POp^{\phi}_{\mathcal{O}}$ but $P \in POp^{\phi}_{\mathcal{O}_t}$, i.e. the process P is prone to timing attack with respect to $\mathcal{O}$ and ϕ. If $\mathcal{O}$ and ϕ is clear from a context we will omit it. Now we define what it means that process can be immunized by a time insertion function.

Definition 8. *We say that process P can be immunized for process opacity with respect to a predicate ϕ over $Actt^\star$ and the observation function $\mathcal{O}$ if for every P', $P \xrightarrow{w} P'$ such that $\phi(P')$ holds and there does not exists P'' such that $P \xrightarrow{w'} P''$ for some w' such that $\mathcal{O}(w) = \mathcal{O}(w')$ and $\phi(P'')$ does not hold, there exist w_t, $w \ll_{\{t\}} w_t$ such that $P \xrightarrow{w_t} P''$ and and there exists P''' and w'', such that $P \xrightarrow{w''} P'''$ such that $\neg\phi(P''')$ holds and $\mathcal{O}(w_t) = \mathcal{O}(w'')$.*

In Fig. 1 process immunization is depicted.

$$\begin{array}{llll}
P & \overset{w}{\Longrightarrow} & \phi(P') & \mathcal{O}(w) \\
 & & & \| \\
P & \overset{w'}{\not\Longrightarrow} & \neg\phi(P'') & \mathcal{O}(w') \\
 & & & \\
P & \overset{w_t}{\Longrightarrow} & \phi(P') & \mathcal{O}(w_t) \\
 & & & \| \\
P & \overset{w''}{\Longrightarrow} & \neg\phi(P''') & \mathcal{O}(w'')
\end{array}$$

Figure 1: Process immunization

Now we will study an existence of time insertion functions. First we need some notations. We begin with observational functions which do not see τ actions.

Definition 9. *We say that observational function $\mathcal{O}$ is not sensitive to τ action iff $\mathcal{O}(w) = \mathcal{O}(w|_{At})$ for every $w \in Act^*$. Otherwise we say that $\mathcal{O}$ is sensitive to τ action.*

Example 2. *Process P, $P = t.R + (a.Q|\bar{a}.Nil) \setminus \{a\}$, cannot be immunized if $\mathcal{O}$ is sensitive to τ action and $\phi(R)$, $\neg\phi(Q)$ hold. An immunization should put a time delay into the trace performed by the right part of the process P but this subprocess cannot perform t action due to the inference rule Pa before communication by means of channel a.*

Lemma 1. *Let P is prone to timing attack with respect to $\mathcal{O}$ and ϕ. Let $\tau \notin L(P)$, P is sequential (i.e. does not contain parallel composition) and $\mathcal{O}$ is static. Then P can be immunized.*

Another problem, which causes that processes cannot be immunized, is related to observation of time, namely, if this observation is context sensitive, as it is stated by the following example.

Example 3. *Process $P, P = h.R.Nil + t.Q.Nil$ cannot be immunized for dynamic $\mathcal{O}$ such that $\mathcal{O}(w.h.t^*.w') = \mathcal{O}(w.w')$, $\mathcal{O}(w.l.w') = \mathcal{O}(w).l.\mathcal{O}(w')$, if w does not end with action h we have $\mathcal{O}(w.t.w') = \mathcal{O}(w).t.\mathcal{O}(w')$, and $\phi(R)$, $\neg\phi(Q)$ hold.*

Now we define time contextuality formally.

Definition 10. *We say that observational function $\mathcal{O}$ is time non-contextual if $\mathcal{O}_t(w) = \mathcal{O}_t(w')$ for every w, w' such that $w \ll_{\{t\}} w'$.*

Proposition 1. *Let process P is prone to timing attacks with respect to ϕ and time non-contextual observation function $\mathcal{O}$ which does not see τ. Then P can be immunized for opacity with respect to timing attacks.*

Proof 1. *The main idea. Let $P \notin POp^{\phi}_{\mathcal{O}}$ but $P \in POp^{\phi}_{\mathcal{O}_t}$. This means that there exists a sequence w and P' such that $P \xrightarrow{w} P'$, $\phi(P')$ holds and there does not exist P'' such that $P \xrightarrow{w'} P''$ for some w' such that $\mathcal{O}(w) = \mathcal{O}(w')$ and $\phi(P'')$ does not hold.*

Suppose that P cannot be immunized, i.e. for every $w_t, w \ll_{\{t\}} w_t$ if $P \xrightarrow{w_t} P'''$, $\phi(P''')$ holds, there does not exist P''''' such that $P \xrightarrow{w''} P''''$ for some w'' such that $\mathcal{O}(w_t) = \mathcal{O}(w''')$. But due to our assumption we have $\mathcal{O}_t(w_t) = \mathcal{O}_t(w)$ and hence it is with contradiction that P is prone to timing attacks.

Corollary 1. Let $\mathcal{O}$ is a static observation function such that $\mathcal{O}(\tau) = \epsilon$ and $\mathcal{O}(t) = t$. Then process P which is prone to timing attacks with respect to ϕ and observation function $\mathcal{O}$ can be immunized for process opacity with respect to timing attacks.

Under some special conditions time insertion functions can be computed effectively a it is stated by the following proposition.

Proposition 2. *Let process P is prone to timing attacks with respect to ϕ and time non-contextual observation function $\mathcal{O}$ which does not see τ. Then P can be immunized for opacity with respect to timing attacks by a m-orwellian insertion function, moreover such one, which can be emulated by finite state process.*

Proof 2. *Sketch. The prove follows an idea from Proposition 3 and Theorem 4.10 and Lemma 4.5.in [13], where insertion functions are modeled by processes run in parallel with P.*

No we define what it means that a predicate is time sensitive.

Definition 11. *We say that predicate ϕ is time sensitive iff whenever $\phi(P)$ holds for P then there exists $n, n > 0$ such that $P \xrightarrow{t^n} P'$ and $\phi(P')$ does not hold.*

Proposition 3. *Let process P is prone to timing attacks with respect to time sensitive predicate $\neg\phi$ and time non-contextual observation function $\mathcal{O}$ which does not see τ. Then P can be immunized for opacity with respect to timing attacks, $\mathcal{O}$ and ϕ.*

Proof 3. *Sketch. By inserting some time actions after performing a sequence we can always reach a state for which ϕ holds and hence P becomes safe with respect to $POp_{\mathcal{O}}^{\neg\phi}$.*

Corollary 2. Let process P is prone to timing attacks with respect ϕ and time non-contextual observation function $\mathcal{O}$ which does not see τ. Let $\neg\phi$ is not time sensitive predicate then P can be immunized for opacity with respect to timing attacks and $\mathcal{O}$ andϕ.

In general, we cannot decide whether process can be immunized as it is stated by the following proposition. Fortunately, it is decidable for the most important static and m-orwellian observation functions.

Proposition 4. *Immunizability is undecidable i.e. it cannot be decided whether P can be immunized for opacity with respect to timing attacks.*

Proof 4. *Sketch. Let T_i represents i-th Turing machine under some numeration of all Turing machines. We start with generalized process from Example 3. Let $P = h.l.R + \sum_{i \in N} t^i.l.Q$. Let $\mathcal{O}(w.h.t^i.w') = \mathcal{O}(w.w')$ whenever T_i halts with the empty tape and $\mathcal{O}(w.h.t^i.w') = \mathcal{O}(w).t^i.\mathcal{O}(w')$ otherwise. It is easy to check that immunization of P is undecidable.*

Proposition 5. *Immunizability is decidable for static and m-orwellian observation function $\mathcal{O}$.*

Proof 5. *According to Proposition 3 it is enough to show that observation function $\mathcal{O}$ is time non-contextual observation function and it does not see τ. Clearly both these properties are decidable for static and m-orwellian observation functions.*

4. Conclusions

We have investigated time insertion functions for timed process algebra which enforce the security with respect to timing attack formalized process opacity. Time insertion functions add some delays to system's behaviour to prevent a timing attack. We study an existence of an insertion function for a given process, given observational function and a predicate over processes. In future work we plan to investigate minimal insertion functions, i.e. such functions which add as little as possible time delays to guarantee process's security with respect to timing attacks. The presented approach allows us to exploit also process algebras enriched by operators expressing other "parameters" (space, distribution, networking architecture, power consumption and so on). Hence we could obtain security properties which have not only theoretical but also practical value. Moreover, we can use similar techniques as in [21] to minimize time, as well as other resources, added to process's behaviour. Moreover, we plan to model both observation functions as well as predicates over processes by processes themselves, to obtain some complexity results as it was done in [13] for trace based variant of opacity.

Acknowledgement

This work was supported by the Slovak Research and Development Agency under the Contract no. APVV-19-0220 (ORBIS) and by the Slovak VEGA agency under Contract no. 1/0778/18 (KATO).

References

[1] P. Ramadge, W. Wonham, The control of discrete event systems, Proceedings of the IEEE 77 (1989) 81–98. doi:10.1109/5.21072.

[2] D. P. Gruska, M. C. Ruiz, Opacity-enforcing for process algebras, in: B. Schlingloff, S. Akili (Eds.), Proceedings of the 27th International Workshop on Concurrency, Specification and Programming, Berlin, Germany, September 24-26, 2018, volume 2240 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2018. URL: http://ceur-ws.org/Vol-2240/paper1.pdf.

[3] P. C. Kocher, Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems, in: N. Koblitz (Ed.), Advances in Cryptology — CRYPTO '96, Springer Berlin Heidelberg, Berlin, Heidelberg, 1996, pp. 104–113.

[4] J.-F. Dhem, F. Koeune, P.-A. Leroux, P. MestrÃ©, J.-J. Quisquater, J.-L. Willems, A practical implementation of the timing attack, volume 1820, 1998, pp. 167–182. doi:10.1007/10721064_15.

[5] H. Handschuh, H. M. Heys, A timing attack on rc5, in: Proceedings of the Selected Areas in Cryptography, SAC '98, Springer-Verlag, Berlin, Heidelberg, 1998, p. 306–318.

[6] A. Hevia, M. Kiwi, Strength of two data encryption standard implementations under timing attacks, ACM Trans. Inf. Syst. Secur. 2 (1999) 416–437. URL: https://doi.org/10.1145/330382.330390. doi:10.1145/330382.330390.

[7] F. Koeune, F. Koeune, J.-J. Quisquater, J. jacques Quisquater, A timing attack against Rijndael, Technical Report, 1999.

[8] D. X. Song, D. Wagner, X. Tian, Timing analysis of keystrokes and timing attacks on ssh, in: Proceedings of the 10th Conference on USENIX Security Symposium - Volume 10, SSYM'01, USENIX Association, USA, 2001.

[9] R. Jacob, J.-J. Lesage, J.-M. Faure, Overview of discrete event systems opacity: Models, validation, and quantification, Annual Reviews in Control 41 (2016) 135–146. URL: https://www.sciencedirect.com/science/article/pii/S1367578816300189. doi:https://doi.org/10.1016/j.arcontrol.2016.04.015.

[10] C. Keroglou, L. Ricker, S. Lafortune, Insertion functions with memory for opacity enforcement, IFAC-PapersOnLine 51 (2018) 394–399. URL: https://www.sciencedirect.com/science/article/pii/S240589631830661X. doi:https://doi.org/10.1016/j.ifacol.2018.06.331, 14th IFAC Workshop on Discrete Event Systems WODES 2018.

[11] C. Keroglou, S. Lafortune, Embedded insertion functions for opacity enforcement, IEEE Transactions on Automatic Control 66 (2021) 4184–4191. doi:10.1109/TAC.2020.3037891.

[12] Y.-C. Wu, S. Lafortune, Enforcement of opacity properties using insertion functions, in: 2012 IEEE 51st IEEE Conference on Decision and Control (CDC), 2012, pp. 6722–6728. doi:10.1109/CDC.2012.6426760.

[13] D. P. Gruska, Security and time insertion, Proceedings of the 23rd Pan-Hellenic Conference on Informatics (2019).

[14] D. P. Gruska, Time insertion functions, in: Proceedings CSMML 2021, to appear, 2021.

[15] D. P. Gruska, Process opacity for timed process algebra, in: A. Voronkov, I. B. Virbitskaite (Eds.), Perspectives of System Informatics - 9th International Ershov Informatics Conference, PSI 2014, St. Petersburg, Russia, June 24-27, 2014. Revised Selected Papers,

volume 8974 of *Lecture Notes in Computer Science*, Springer, 2014, pp. 151–160. URL: https://doi.org/10.1007/978-3-662-46823-4_13. doi:10.1007/978-3-662-46823-4_13.

[16] R. Milner, Communication and Concurrency, Prentice-Hall, Inc., USA, 1989.

[17] R. Focardi, R. Gorrieri, F. Martinelli, Information flow analysis in a discrete-time process algebra, in: Proceedings 13th IEEE Computer Security Foundations Workshop. CSFW-13, 2000, pp. 170–184. doi:10.1109/CSFW.2000.856935.

[18] R. Gorrieri, F. Martinelli, A simple framework for real-time cryptographic protocol analysis with compositional proof rules, Science of Computer Programming 50 (2004) 23–49. doi:10.1016/j.scico.2004.01.001.

[19] J. W. Bryans, M. Koutny, P. Y. Ryan, Modelling opacity using petri nets, Electronic Notes in Theoretical Computer Science 121 (2005) 101–115. URL: https://www.sciencedirect.com/science/article/pii/S1571066105000277. doi:https://doi.org/10.1016/j.entcs.2004.10.010, proceedings of the 2nd International Workshop on Security Issues with Petri Nets and other Computational Models (WISP 2004).

[20] J. Bryans, M. Koutny, L. Mazare, P. Ryan, Opacity generalised to transition systems, volume 7, 2008, pp. 421–435. doi:10.1007/11679219_7.

[21] Y. Ji, X. Yin, S. Lafortune, Enforcing opacity by insertion functions under multiple energy constraints, Automatica 108 (2019) 108476. URL: https://www.sciencedirect.com/science/article/pii/S0005109819303243. doi:https://doi.org/10.1016/j.automatica.2019.06.028.